Theoretical Computer Science 133 (1994) 421–442
Elsevier

Aggregate operations in the information source tracking method

Fereidoon Sadri*

Department of Computer Science, Concordia University, Montreal, Canada

Abstract

Sadri, F., Aggregate operations in the information source tracking method, Theoretical Computer Science 133 (1994) 421–442.

The *Information Source Tracking* method, IST, is an approach to the management of uncertain and imprecise data in database systems. In this paper we study the processing of queries involving aggregate operations *min*, *max*, *sum*, *count*, and *average* in the IST method. The problems discussed include producing all possible outcomes of an aggregate query and their probabilities; determining the probability of a specific outcome; finding the largest (or smallest) possible outcome; determining whether the outcome could be greater than or equal to (or less than or equal to) a given value; and finding the expected-value of the outcome of an aggregate query. We present algorithms for the evaluation of aggregate queries, and show that some of these problems are NP-complete, and hence highly unlikely to have efficient algorithms.

1. Introduction

The *Information Source Tracking* method, IST, is an approach to the management of uncertain and imprecise data in database systems [11]. In this paper we study the processing of queries involving aggregate operations in the IST method.

The main idea behind IST is that database information is supplied, or confirmed, by information sources. The accuracy of data is modeled by the reliability of the contributing information sources. The IST method uses an extended relational model. The identity(ies) of contributing information source(s) are stored along with each tuple in the database. Extended relational algebra operations are provided that manipulate data as well as information regarding contributing information sources.

Correspondence to: F. Sadri, Department of Computer Science, Concordia University, Montreal, H3G 1M8, Canada.

*Supported by grants from the Natural Sciences and Engineering Research Council of Canada (NSERC), and Fonds pour la Formation de Chercheurs et l'Aide a la Recherche (FCAR) of Quebec.

SSDI 0304-3975(94)00050-S

In response to a query, the system provides the answer, and also identifies information sources contributing to each answer. More precisely, the IST method identifies the exact conditions under which an answer to a query is valid. Given a quantitative measure of the reliability of information sources, the reliability of answers to queries can be calculated. An SQL interface based on the IST model has been implemented which supports a subset of the SQL query language [4]. In this paper we study how the aggregate operations, *min*, *max*, *sum*, *count* and *average*, can be implemented in the IST model.

The rest of this paper is organized as follows. Section 2 is a background review of the IST method, the *alternate worlds semantics* model of the IST, and the reliability calculation algorithms. In Section 3 we introduce aggregate operations in the IST model, and present an algorithm to calculate all possible answers to an aggregate query and their probabilities. Section 4 is devoted to a discussion of different formulations of the intended meaning of aggregate operations in the IST model, and their complexity. The formulations include

- Determining the probability of a specific outcome of an aggregate query.
- Finding the largest (or smallest) possible outcome.
- Determining whether the outcome could be greater than or equal to (or less than or equal to) a given value.
- Finding the expected-value of the outcome of an aggregate query.

We show that some of these problems are NP-complete, and hence highly unlikely to have an efficient algorithm. In Section 5 we present algorithms for the calculation of expected-values of answers to *sum* and *count* queries in the IST model. Algorithms for the processing of *min* and *max* queries for various formulations are also presented in Section 5. Finally, concluding remarks are presented in Section 6.

2. A review of the information source tracking method

The *Information Source Tracking* method, IST, was introduced in [11], and a semantic interpretation for IST was presented in [12]. The issue of consistency in the IST method, ie., how to handle conflicting information supplied by different information sources, was studied in [13], and the efficiency of query processing and reliability calculation algorithms of IST was studied in [14]. In this section we briefly review some of the concepts from [11, 12] needed to discuss aggregate operations in the IST method. The model represented in this paper is more general than that of the original papers [11, 12], nevertheless most of the previous discussions and results can be shown to be valid for the more general model.

2.1. Information source tracking

The idea behind IST is simple: we record, for each tuple in a relation in the database, the source of the tuple (called the *confirming* or *contributing* information

source). Query processing is performed using extended relational algebra operations that manipulate contributing source data in addition to traditional database data. The answer to a query is a relation (similar to a traditional relational database system), but there is also precise information regarding contributing information sources for each tuple in the answer. In this way, we can determine the conditions under which a tuple in the answer is valid. If a quantitative measure of the reliabilities of the information sources is available (supplied by users and/or database administrator), then a quantitative measure of validity can be calculated for each tuple in the answer. In what follows we provide precise definitions of the IST method.

Definitions. The IST model is based upon an extended relational model discussed below. An *extended relation scheme* is a set of attributes $\{A_1, \ldots, A_n, I\}$, where $A_1, \ldots, A_n$ are regular attributes, and I is a special attribute, called the *information source attribute* (*source attribute*, for short). Each attribute, A_i has a *domain* of values D_i, $i=1, \ldots, n$. The domain of the source attribute I, denoted by D_I, is the set of vectors of length k with $\{0, -1, +1, \top\}$ elements, that is, $D_I = \{(a_1 \cdots a_k) \; a_i \in \{0, -1, +1, \top\}, \; i=1, \ldots, k\}$, where k is the number of information sources. An element of D_I is called an *information source vector* (*source vector*, for short). We require that a source vector should have at least one nonzero element. We also permit special source vectors T and F, intended for tuples known to be *true* and *false*, respectively.

A *tuple* on the (extended) scheme $R = \{A_1, \ldots, A_n, I\}$ is an element of $D_1 \times \cdots \times D_n \times D_I$. We usually write $t@u$ to denote a tuple on the extended scheme R, where t is the value of the tuple corresponding to the regular attributes $A_1, \ldots, A_n$, and u is the value of the tuple corresponding to the source attribute I. We call t a *pure tuple*, and u is the *source vector* corresponding to t. A *relation instance* (*relation* for short) r on the scheme R is a set of tuples on R.

A source vector u for a tuple $t@u$ identifies sources that contribute to t. An entry of $a_i = 0$ in a source vector specifies that source s_i has not contributed to the corresponding tuple, while $a_i = \top$ indicates that source s_i is inconsistent with respect to the corresponding tuple. This case happens if the validity of a tuple is dependent on a source s_i to be correct as well as incorrect at the same time.

We do not permit contradictory information from an information source. Hence the source vectors associated with tuples in base (stored) relations will not contain $\top$ elements. The $\top$ entries may appear in derived relations, such as answers to queries and views. For example, if a source s_i conforms facts f_1 and f_2, and some derived fact f is valid if f_1 holds but f_2 does not, then the source s_i is inconsistent with respect to f. The earlier IST presentations [11, 12] did not use the $\top$ element in source vectors, rather a vector of zeros was used to indicate inconsistency. The introduction of $\top$ element provides a precise presentation of inconsistency, resulting in a cleaner semantics.

Intuitively, a tuple $t@u$ is valid if all sources having a $+1$ entry in u are correct, and all those having a -1 entry in u are incorrect. Of course, if u has a $\top$ element, then it is

not valid since there is an information source which is inconsistent with respect to t. We will make this notion precise further below, when we discuss the *expression* corresponding to a tuple.

Usually, each tuple of a stored database relation (sometimes called a *base* relation), is supplied by a single information source. Hence, for base relations, each extended tuple $t@u$ has a source vector u with only one $+1$ element. If a tuple t is supplied independently by several sources $s_i, s_j, \ldots$, then the extended tuples $t@u_i$, $t@u_j, \ldots$, will be in the base relation where each $u_i, u_j, \ldots$, has a single $+1$ element. Once we apply relational algebra operations, for example to produce answers to a query, we may obtain tuples having source vectors with possibly several nonzero $(-1, +1, \top)$ elements. The extended relational algebra operations are discussed in the next subsection. The -1 entries arise from the set difference operation (or the NOT EXISTS construct of SQL). Note that there can be more than one source vector associated with a pure tuple t in a relation, i.e., $t@u_1, \ldots, t@u_p$, $p \geqslant 1$, can be in r.

The interpretation of source vectors is made precise by introducing the *expression* corresponding to a pure tuple t as follows. We associate a Boolean variable f_i with each information source s_i, $i=1, \ldots, k$. Let $u=(a_1, \ldots, a_k)$ be a source vector, then the set of information sources $S^+ = \{S_i \mid a_i = +1 \text{ or } a_i = \top\}$ are contributing positively to u, while the set of information sources $S^- = \{S_i \mid a_i = -1 \text{ or } a_i = \top\}$ are contributing negatively. Note that the case of inconsistent information source, $a_i = \top$, is treated as the source s_i is contributing positively as well as negatively to the corresponding tuple. The expression $e(u)$ corresponding to the source vector $a=(a_1, \ldots, a_k)$, is

$$e(u) = \bigwedge_{s_i \in S^+} f_i \bigwedge_{s_i \in S^-} \neg f_j.$$

Note that $e(u)=\mathit{false}$ if $u_i = \top$ for at least one i, $1 \leqslant i \leqslant k$. Hence, a tuple with an inconsistent information source is not valid. The expressions corresponding to the special source vectors T and F are *true* and *false*, respectively.

The expression $e(x)$ corresponding to a set x of source vectors is

$$e(x) = \bigvee_{u \in x} e(u).$$

Finally, the expression corresponding to the pure tuple t in an extended relation r is defined as

$$e(t) = e(x),$$

where $t@x \in r$.

We can regard the expression corresponding to a tuple t in a relation r as a propositional logic expression, where $f_1, \ldots, f_k$ represent Boolean variables. A truth assignment $f_i = \mathit{true}$ is interpreted as "information source s_i is correct", otherwise, $f_i = \mathit{false}$ which indicates s_i is incorrect. The truth value of $e(t)$ is a function of the truth values of $f_1, \ldots, f_k$, and indicates whether t is a valid tuple ($e(t) = \mathit{true}$), or an invalid tuple ($e(t) = \mathit{false}$).

The expression corresponding to a tuple t in a relation r can also be used to derive probabilistic information about t, i.e., given probabilities for correctness of sources $s_1, \ldots, s_k$, we can calculate the probability of the validity of t [11, 14].

2.2. *Source vector operations*

In the IST model query processing is achieved using extended relational algebra operations. These extended operations operate on the regular data as their standard counterparts (namely, selection, projection, Cartesian product, natural join, union, and set difference). In addition, the extended operations also operate on source vectors to produce source vectors for the tuples in the result that precisely determine under what conditions the associated tuple is valid. The extended relational algebra operations are defined in terms of source vector operations *s-conjunction*, *s-disjunction*, and *s-negation*. We will discuss these operations in this section.

We can regard the information source constants $\{0, -1, +1, \top\}$ as a lattice structure, with the top and bottom elements $\top$ and 0, respectively. The partial order among the elements are as follows: $0 \prec 1 \prec \top$ and $0 \prec -1 \prec \top$. The bottom element, 0, can be considered as designating *under specified*, and the top element, $\top$, designates *over specified* (or *inconsistent*).

Given two sources vectors $v=(a_1 \cdots a_k)$ and $w=(b_1 \cdots b_k)$, their *s*-conjunction $u=v \overset{s}{\wedge} w$ is a source vector $u=(c_1 \cdots c_k)$ obtained as follows:

$$c_i = lub(a_i b_i),$$

where *lub* is the least upper bound with respect to the lattice of information source constants, enumerated in Table 1 for convenience.

Table 1

a_i	b_i	$c=lub(a_i, b_i)$
0	0	0
0	−1	−1
0	+1	+1
0	$\top$	$\top$
−1	0	−1
−1	−1	−1
−1	+1	$\top$
−1	$\top$	$\top$
+1	0	+1
+1	−1	$\top$
+1	+1	+1
+1	$\top$	$\top$
$\top$	0	$\top$
$\top$	−1	$\top$
$\top$	+1	$\top$
$\top$	$\top$	$\top$

The cases where one or both operands are the special source vectors T or F are handled in the obvious way, namely:

$$F \overset{s}{\wedge} v = v \overset{s}{\wedge} F = F \quad \text{for all source vectors } v, \text{ and}$$

$$T \overset{s}{\wedge} v = v \overset{s}{\wedge} T = v \quad \text{for all source vectors } v.$$

The conjunction of two sets of source vectors x and y is performed as follows

$$x \overset{s}{\wedge} y = \{v \overset{s}{\wedge} w \mid v \in x \text{ and } w \in y\}.$$

The disjunction of two sets of source vectors x and y, written $x \overset{s}{\vee} y$, is their union

$$x \overset{s}{\vee} y = x \cup y.$$

The negation of a source vector $v = (a_1 \cdots a_k)$, written $\overset{s}{\neg} v$, is defined as follows: let u_i denote the source vector $(b_1 \cdots b_k)$ where $b_i = +1$ and $b_j = 0$, for all $j \neq i$, and similarly let w_i denote the source vector $(b_1 \cdots b_k)$ where $b_i = -1$ and $b_j = 0$, for all $j \neq i$, then

$$\overset{s}{\neg} v, = \{u_i \mid a_i = -1 \text{ or } a_i = \top\} \cup \{w_i \mid a_i = +1 \text{ or } a_i = \top\}.$$

The negation of the special source vectors T and F are F and T, respectively.

The negation of a set $x = v_1, \ldots, v_q$ of source vectors is calculated as follows:

$$\overset{s}{\neg} x = (\overset{s}{\neg} v_1) \overset{s}{\wedge} (\overset{s}{\neg} v_2) \overset{s}{\wedge} \cdots \overset{s}{\wedge} (\overset{s}{\neg} v_q)$$

The following results were proven in [11]. Note that the definition of source vectors in this paper is more general than that of [11], but these results can be easily extended to the more general case. In the following x, y, and z are sets of source vectors, and $e(x)$, $e(y)$, and $e(z)$ are their corresponding expressions, respectively.

Theorem 1. *Let* $z = x \overset{s}{\wedge} y$. *Then* $e(z) = e(x) \wedge e(y)$.

Theorem 2. *Let* $z = x \overset{s}{\vee} y$. *Then* $e(z) = e(x) \vee e(y)$.

Theorem 3. *Let* $z = \overset{s}{\neg} x$. *Then* $e(z) = \neg e(x)$.

2.3. Extended relational algebra operations

Now we can summarize the extended relational algebra operations: extended selection, projection, and union are similar to their regular counterparts

$$\sigma_C(r) = \{t @ u \mid t @ u \in r, \text{ and } t \text{ satisfies condition } C\},$$

$$\Pi_X(r) = \{t[X] @ u \mid t @ u \in r\},$$

$$r \cup s = \{t @ u \mid t @ u \in r \text{ or } t @ u \in s\}.$$

Note that an implicit s-disjunction operation takes place for source vectors with the same pure component in the union, and for source vectors with the same X component in the projection. We shall also note that the information source attribute I is not visible to users, and can not be referenced (e.g., in the condition of a selection or in the attribute set of a projection).

Intersection, Cartesian product, and natural join are defined using the s-conjunction operation for source vectors:

$$r \cap S = \{t@(u_1 \overset{s}{\wedge} u_2) | t@u_1 \in r \text{ and } t@u_2 \in s\},$$

$$r \times s = \{(t_1 \cdot t_2)@(u_1 \overset{s}{\wedge} u_2) | t_1@u_1 \in r, \text{ and } t_2@u_2 \in s\},$$

$$r \bowtie s = \{(t_1 \otimes t_2)@(u_1 \overset{s}{\wedge} u_2) | t_1@u_1 \in r, t_2@u_2 \in s, \text{ and } t_1 \text{ and } t_2 \text{ join}\}.$$

where $t_1 \cdot t_2$ indicates the concatenation of t_1 and t_2, and $t_1 \otimes t_2$ indicates the join of t_1 and t_2, i.e., the concatenation of t_1 and t_2 with the removal of duplicate values of common attributes. Two tuples t_1 and t_2 join if they have the same values for the common attributes. Note that t_1 and t_2 are *pure* tuples and do not contain values for the information source attribute I. In other words, we should try to match the information source vectors when joining.

Finally, set difference uses s-negation and s-conjunction of source vectors

$$r - s = \{t@x | t@x \in r, \text{ and the pure tuple } t \text{ does not appear in } s, \text{ or,}$$
$$t@y \in r, t@z \in s, \text{ and } x = y \overset{s}{\wedge} (\overset{s}{\neg} z)\}$$

2.4. The alternate worlds semantics

The *alternate worlds model* was presented in [12] to provide a semantic interpretation for IST. The idea is similar to the notions of "representation", "possibility functions", "alternate worlds", and "possible worlds" used by researchers in databases and artificial intelligence [1, 3, 6, 7, 9, 10]. It was shown in [12] that the extended relational algebra operations of IST are *precise* under the alternate worlds interpretation. The reliability calculation algorithms were also shown to be correct under this interpretation. Here we briefly review the alternate worlds semantics.

An extended relation *represents* a set of (regular) relations. This set of regular relations is called the *alternate world* of the extended relation. We give precise definitions below.

Definitions. Given a relation r on the (extended) scheme R, r can be written as $r = \{t_1@x_1, \ldots, t_n@x_n\}$, where $x_1, \ldots, x_n$ are sets of source vectors. We define r^* as a function from the set of subsets of information sources S to the set of (regular) relations *Rel* on the scheme $R - \{I\}$, that is,

$$r^*: 2^S \rightarrow Rel. \tag{1}$$

Let $Q \subseteq S$ be a set of information sources. Assign truth value "true" to sources in Q, and "false" to other sources. (We will denote this truth assignment by *truth*(Q).) Then

$$r^*(Q) = \{t | t@x \in r \text{ and } e(t) = true \text{ under } truth(Q)\}, \tag{2}$$

where $e(t) = e(x)$ is the expression corresponding to t in r.

An extended relation r *represents* the function r^*. The set of (regular) relations $r^*(Q)$, $Q \subseteq S$, is called the *alternate world* of r. Informally, an extended relation r represents the set of (regular) relations consisting of those tuples that would be valid if the information sources in Q were correct and all other information sources were incorrect, for all $Q \subseteq S$.

Example 1. Consider the relation *employee* of Fig. 1.

Employee	Salary	I
a	40000	1 0
b	60000	0 1

Fig. 1. The employee relation.

The alternate world of *employee* consists of four relations, the empty relation $r_1 = \phi$, corresponding to the empty set of information sources, plus relations r_2, r_3, and r_4 of Fig. 2, corresponding to $\{s_1\}$, $\{s_2\}$, and $\{s_1, s_2\}$, respectively.

Employee	Salary	Employee	Salary	Employee	Salary
a	40000	b	60000	a	40000
				b	60000

Fig. 2. Relations r_2, r_3, and r_4.

In [12] we prove that extended relational algebra operations introduced in [11] are *precise* under the alternate worlds semantics. That is, informally, the extended operations applied to extended relations produce exactly the same result as the regular operations applied to the alternate world of the corresponding extended relations.

We also make the observation here that if a source vector u has a $\top$ element, then $e(u) = false$ under $truth(Q)$ for all $Q \subseteq S$. It follows that if $u \in x$ and u has a $\top$ element, then $e(x) = e(x - \{u\})$ under $truth(Q)$ for all $Q \subseteq S$. Hence in an extended relation r, we can remove any extended tuple $t@u$ having a source vector u with a $\top$ element without affecting the alternate world of r. We call an extended relation r with no $\top$ element in its source vectors an *overspecification-free* relation. Since the "meaning" of an extended relation was defined as its alternate worlds interpretation, we can eliminate the $\top$ elements according to the above observation without changing the semantics of an extended relation. Henceforth, we will assume that our relations are overspecification-free.

2.5. Reliability calculation

In the IST model users (or database administrator) can provide reliability figures for information sources. The *reliability* of a source s_i is defined as the probability that data confirmed by s_i is correct, and is denoted by p_i. The query processing in a database system based on IST is carried out using extended relational algebra operations. Once an answer is obtained for a query, the reliability of each tuple in the

answer can be calculated as a function of the contributing information sources reliabilities.

Two algorithms for the calculation of the reliabilities of the tuples in the answer to a query were presented in [11] and proven correct under alternate worlds semantics in [12]. Here, we briefly review Algorithm 2 from [11].

The algorithm is based on the conversion of source vectors into *disjunctive normal form*. For example a source vector (1 0 0) is equivalent to the set of source vectors $\{(1\ 1\ 1), (1\ -1\ 1), (1\ 1\ -1), (1\ -1, -1)\}$. We also assume that the source vectors with $\top$ elements have been eliminated according to the observation made at the end of the previous subsection. Let $t@\{v_1, \ldots, v_q\}$ be all the tuples with pure component t, where v_i's are in disjunctive normal form. The reliability of t, denoted by $re(t)$, is calculated as

$$re(t) = re(t@v_1) + \cdots + re(t@v_q), \tag{3}$$

where, for a single source vector u,

$$re(t@u) = \prod_{s_i \in S^+(u)} p_i \prod_{s_i \in S^-(u)} (1 - p_i), \tag{4}$$

where $S^+(u)$ and $S^-(u)$ are the set of information sources contributing to $t@u$ positively and negatively, respectively. That is, for $u = (a_1 \cdots a_k)$,

$$S^+(u) = \{s_i | a_i = +1,\ i = 1, \ldots, k\},$$

$$S^-(u) = \{s_i | a_i = -1,\ i = 1, \ldots, k\}.$$

3. Aggregate operations in IST

The aggregate operations, such as *min*, *max sum*, *count*, and *average* in SQL, are needed frequently in many applications. In a regular database, where no uncertainty exists, the processing of queries involving aggregate operations is straightforward. The algorithms for these operations have a linear time performance in the size of the relation involved (except when an index is available, where some operations, such as *min* and *max*, can be performed more efficiently).

The situation becomes much more complicated in the presence of uncertain and imprecise data. In this section we discuss the meaning of the aggregate operations in the IST model, and present a brute-force algorithm for processing such operations.

Our starting point will be the alternate worlds semantics of the IST model. Recall that if k information sources $s_1, \ldots, s_k$ contribute to an extended relation r, then the alternate world of r consists of up to 2^k regular relations. Each subset Q of the set of information sources, $Q \subseteq \{s_1, \ldots, s_k\}$, defines a regular relation represented by r, but some of these regular relations may be identical. A possible interpretation of the result of an aggregate operation on an extended relation r is the list of the possible results of the operation carried out on the regular relations represented by r, together with their probabilities. More precisely, we define the result of an aggregate operation *agg* on an

extended relation r with respect to an attribute A as a function agg^* from the set of subsets of $S=\{s_1, \ldots, s_k\}$ to values, $agg^*: 2^S \rightarrow R$, where

$$agg^*_A(Q) = agg_A(r^*(Q))$$

for all $Q \subseteq S$, where agg_A applied to a regular relation is the classical aggregate operation, and $r^*(Q)$ is the regular relation represented by r corresponding to $Q \subseteq S$.

Example 2. Consider the extended relation *employee* of Example 1 (Fig. 1). The relations in the alternate world of *employee* consists of the empty relation $r_1 = \phi$ and the relations r_2, r_3, and r_4 shown in Fig. 2. The query $\text{sum}_{salary}(employee)$, or in SQL

select sum(salary)
from employee

can be evaluated against the four regular relations r_1 to r_4 in the alternate world of r. The answer would be 0, 40 000, 60 000, and 100 000, respectively.

Given reliability figures for the information sources, we can associate a probability with each regular relation in the alternate world of an extended relation [12]. These probabilities can be associated, in a straightforward manner, with the results of an aggregate operation on these regular relations. For example, if the reliabilities of the two information sources in the above example were 90% and 80%, respectively, then the probabilities associated with the answers, 0, 40 000, 60 000, and 100 000, would be 2%, 18%, 8%, and 72%, respectively.

3.1. An algorithm for aggregate operations in the IST model

A brute-force algorithm to enumerate the answers of an aggregate operation on the relations in the alternate world of an extended relation by obtaining the alternate world relations and processing the aggregate operation on each of them is clearly inefficient. There are an exponential number of relations (exponential in the number of information sources) in the alternate world of an extended relation in the worst case, and the brute-force algorithm will have an exponential time complexity. In fact, the size of the answer, which is exponential in the worst case eliminates any hopes of finding an efficient algorithm. Note that the number of tuples in an extended relation can be linear in the number of information sources, which means the size of the answer is exponential in the size of the relation. Further, we will prove in the next section that even some simplified problems involving aggregate operations in the IST model are NP-complete and hence unlikely (unless P = NP) to have efficient algorithms. In what follows, we briefly describe an algorithm to enumerate all possible answers to an aggregate query and their probabilities with the help of an example. The algorithm is based on the expansion of the information source vectors to standard forms discussed in [11]. For example, a vector (0 1 0) will be expanded to the set $\{(-1\ 1\ -1), (1\ 1\ -1), (-1\ 1\ 1), (1\ 1\ 1)\}$. We are also assuming that the source vectors having a $\top$ element have been eliminated according to the observation of Section 2.4.

Example 3. Let us expand every tuple in the *employee* relation of Fig. 1 and group the tuples according to the source vectors as seen in Fig. 3.

Employee	Salary	I	
b	60000	−1	1
a	40000	1	−1
a	40000	1	1
b	60000	1	1

Fig. 3. The expanded employee relation.

Obviously, each group corresponds to one alternate world relation. The aggregate operation can be applied to each group separately to obtain the answers. We obtain for our example the details in Fig. 4.

Sum	I	
60000	−1	1
40000	1	−1
100000	1	1

Fig. 4. The sum of salaries relation.

The probability associated with each answer can be obtained directly from the associated source vectors [11]. Any source vector combination that does not appear in the expanded relation corresponds to the empty relation in the alternate world.

The complexity of this algorithm is $O(2^k \times n)$, where k is the number of information sources, and n is the size of the relation. Note that if the number of information sources is fixed (constant) then the algorithm has a linear complexity in the size of the input.

4. Other formulations, and intractability results

As we saw in the previous section providing all possible answers to an aggregate query in the IST model results in algorithms with an exponential time complexity. In this section we will first discuss various questions for aggregate operations, and then

study the complexity for some of them. We will show that some queries that may seem simple in the first sight turn out to be NP-complete, and hence it is unlikely that an efficient algorithm exists for them.

In the following $agg_A(r)$ designates the operation *agg*, one of the usual aggregate operations, on the attribute A of the extended relation r. We will also assume that source vectors having $\top$ elements have been eliminated from the relation according to the observation of Section 2.4.

Instead of listing all possible results of an aggregate operation, we might be interested in the following questions:

(1) Given a constant c, what is the probability associated with the result of the aggregate operation to be equal to c. For example, in the employee relation of previous sections, we may ask the question "What is the probability for the sum of salaries to be 100 000?" The answer would be 72% for our running example.

(2) Find the largest (or smallest) possible value of the result of the aggregate operation. For our running example, the largest sum of salaries is 100 000.

(3) Is it possible for the result to be greater than or equal to (or less than or equal to) a given value? Note that an answer to the previous question also provides an answer to this question, but the converse is not true.

(4) Find the expected value of the result. For our running example, the expected value of sum of salaries is obtained as

$$0 \times 0.02 + 40\,000 \times 0.18 + 60\,000 \times 0.08 + 100\,000 \times 0.72 = 84\,000.$$

In some cases we would be interested to obtain a "normalized" expected value. That is, we only consider nonempty relations in the alternate world as meaningful, and distribute the probability associated with the empty relation over the nonempty ones. This can be accomplished by dividing the probabilities associated with nonempty relation by their sum (which is equal to one minus the probability of the empty relation). For our example, the normalized expected value of sum of salaries is $84\,000/0.98 = 85\,714$.

To show the usefulness of these formulations let us discuss some examples. Consider the process of budget planning by a government for the approaching fiscal year. It has (uncertain) predictions of its revenues and expenses. A government that is determined to eliminate its deficit will use the *smallest sum* of its (predicted) revenues, and the *largest sum* of its (predicted) expenses to balance the budget. A more pragmatic government will probably use the *expected sum* of its revenues and expenses for budget planning.[1]

As another example, to determine the operating temperature range of a critical system, such as those used in space aircrafts, one should use the *smallest min* and *largest max* of the predicted temperatures to reduce the chances of malfunction.

[1] Unfortunately, most governments use the *largest sum* of predicted revenues and the *smallest sum* of predicted expenses for budget planning.

4.1. Intractability results

In this section we will show that some of the formulations discussed above are NP-complete. The first problem to study is the probability of the result of for aggregate operations *sum*, *count*, and *average* is equal to a given value. In fact, we will show that a simpler problem, which we will call problem P1, is NP-complete. First we will consider the problem for the aggregate operation *sum*, P1-SUM. The cases for *count* and *average* follow.

4.1.1. Problem P1-SUM

Given an extended relation r, determine whether the probability of $sum_A(r)=c$ is nonzero, where c is a constant, and A is an attribute of r.

Theorem 4. *Problem* P1-SUM *is* NP-*complete.*

Proof. Given an extended relation r, we can nondeterministically guess a subset Q of the information sources, obtain the corresponding regular relation $r^*(Q)$ in the alternate world of r, and check to see whether $sum_A(r^*(Q))=c$. Hence the problem P1-SUM is in NP. The proof of completeness is by reducing the three satisfiability (3SAT) problem [2, 5] to P1-SUM. Let $C=\{c_1, c_2, \ldots, c_m\}$ be an instance of the 3SAT problem, where each c_i is a disjunctive clause containing three literals over the set of Boolean variables $U=\{u_1, u_2,, \ldots, u_n\}$. We will construct an extended relation r over the scheme $R=\{A, B, I\}$ with $3\times m$ tuples. There are n information sources $s_1, s_2, \ldots, s_n$ corresponding to the Boolean variables. The reliability of each information source is 50% (any value other than 0 and 100% is acceptable). Each clause c_i gives rise to 3 tuples in r, one for each literal in c_i. If a literal L in c_i is a (nonnegated) variable u_k, then the source vector of the corresponding tuple in r has a $+1$ for s_k and zeros for all other information sources. If the literal is a negated variable, then the corresponding source vector has a -1 for the corresponding information source, and zeros for the rest. The value of the attribute A is the same for all the tuples, equal to 1. The value of the attribute B for each tuple generated by a clause c_i is set to i. Hence, for each $1\leqslant i\leqslant m$ there are 3 tuples in r with their B-value equal to i. This construction is clearly polynomial in the size of the 3SAT instance.

We now claim that the probability of $sum_A(r)=m$ is nonzero if and only if C is satisfiable. It is simple to see that $sum_A(r)=m$ is nonzero iff at least one regular relation in the alternate world of r contains m tuples, which can happen iff all the pure tuples in r are valid under the truth assignment $truth(Q)$ for a subset Q of the information sources. The same truth assignment applied to the Boolean variables designates a satisfying assignment for the 3SAT instance C. □

The problems P1-COUNT and P1-AVERAGE are similar to P1-SUM except that the aggregate operation is *count* and *average*, respectively. These problems are also

NP-complete. The proof is similar to the proof for P1-SUM. We will state the theorem and just sketch the proofs below.

Theorem 5. *The problems* P1-COUNT *and* P1-AVERAGE *are* NP-*complete.*

Proof. For P1-AVERAGE the extended relation r is constructed as it was in the case of P1-SUM, except that the A attribute does not have the same value for all tuples. For example, we can make the 3 tuples corresponding to the clause c_1 have an A-value of 2, and the remaining tuples have an A-value of 1. Then $average_A(r)=(m+1)/m$ has a nonzero probability if and only if the 3SAT instance C is satisfiable.

For P1-COUNT the extended relation needs only the B attribute, and $count(r)=m$ has a nonzero probability if and only if the 3SAT instance C is satisfiable. □

4.1.2. Problems P2 and P3

Problem P2 is concerned with finding the largest (or smallest) value of the result of an aggregate operation. Problem P3 asks the question whether the result is greater than or equal to (or less than or equal to) a given value. We can show that these problems are NP-complete for *sum*, *count*, and *average* operations. The proof is different for different formulations. In what follows, we will use P′2 for the "largest", and P″2 for the "smallest" formulations of P2. Similarly, P′3 and P″3 refer to the "greater than or equal" and "less than or equal" versions of the P3 problem.

Theorem 6. *Problems* P′2-SUM, P′2-COUNT, P′3-SUM, *and* P′3-COUNT *are* NP-*complete.*

Proof. The same reduction used for the proofs of P1-SUM and P1-COUNT can also be applied here. Let r be the extended relation constructed in the proof of NP-completeness of P1-SUM. Then the largest $sum_A(r)$ is equal to m if and only if the corresponding instance of the 3SAT problem is satisfiable. The proof for P′2-COUNT is also similar.

The same proof as above can be used for P′3-SUM and P′3-COUNT problems. The only difference is that $sum_A(r)\geqslant m$ if and only if the corresponding instance of the 3SAT problem is satisfiable. □

Theorem 7. *The problems* P″2-SUM, P″2-COUNT, P″-SUM, *and* P″-COUNT *are* NP-*complete.*

Proof. We will only sketch the proof here, and leave the details to the readers. Given an instance $C=\{c_1,\ldots,c_m\}$ of the 3SAT problem, Let F be the (propositional) formula associated with C. We can use DeMorgan's rules to obtain $\neg F$ as a disjunction of conjuncts, and then build an extended relation r with m tuples, where each tuple has a source vector with an expression equal to a conjunct. The other attributes are similar to the cases for P1-SUM and P1-COUNT, respectively. Then the smallest sum

(over attribute A) or count for r is zero if and only if there is a truth assignment that makes all the conjuncts in $\neg F$ false, and hence if and only if the original 3SAT problem is satisfiable. Hence P″ 2-SUM and P″2-COUNT problems are NP-complete.

For the P″3-SUM, and P″3-COUNT problems the questions to ask is whether it is possible that $sum_A(r) \leqslant 0$, and $count(r) \leqslant 0$, respectively. □

Theorem 8. *The problems* P′2-AVERAGE, P″2-AVERAGE, P′3-AVERAGE, *and* P″3-AVERAGE *are NP-complete.*

Proof. The proof is similar to the previous proof, and we will only give a sketch. The negation of the (3SAT instance) formula F and the corresponding (extended) relation r are constructed as before. We also add one more information source, say s_0, and one more tuple t, with a source vector that has a+1 for s_0 and zeros for the remaining sources. For P′2-Average, namely, the largest possible average problem, the value of the attribute A is chosen to be large for the tuple t, and small for the other tuples. It is easy to show that the largest $avg_A(r)$ is equal to $t(A)$ if and only if F is satisfiable. For P″2-AVERAGE, we only need to make $t(A)$ small, and the rest of A-values large. Then the smallest $avg_A(r) = t(A)$ if and only if F is satisfiable. Hence P2-AVERAGE problems are NP-complete.

For the P′3-AVERAGE, and P″3-AVERAGE problems the question to ask is whether it is possible that $avg_A(r) \geqslant t(A)$ and $avg_A(r) \leqslant t(A)$, respectively. □

5. Algorithms

In the previous sections we showed that the problem of listing all answers to an aggregate query in the IST model has an exponential time complexity. Further, we proved that a number of simpler problems, which we designated by problems P1, P2, and P3, are NP-complete for *sum*, *count*, and *average* operations, and hence it is highly unlikely (unless P = NP) that an efficient algorithm can be found for these problems. In this section we present algorithms for some of the remaining problems.

5.1. Determining the expected value of aggregate queries

The expected value of the *sum* and *count* operations can be calculated directly (rather than enumerating alternate world relations.) This was the fourth problem on our list in the previous section, and hence we will call it problem P4.

5.1.1. Algorithm P4-SUM

Given an extended relation $r = \{t_1 @ x_1, t_2 @ x_2, \ldots, t_n @ x_n\}$, where t_i is a pure tuple, and x_i is the set of information source vectors corresponding to t_i, the expected value of $sum_A(r)$ can be calculated as

$$\sum_{i=1}^{n} (t_i(A) \times re(t_i)), \tag{5}$$

where $re(t)$ is the reliability of tuple t calculated by one of the algorithms of [11] (A summary of one of the algorithms was given in Section 2.5.)

Theorem 9. *Algorithm* P4-SUM *correctly calculates the expected value of* $sum_A(r)$.

Proof. The expected value of $sum_A(r)$, by definition, is

$$\sum_{Q \subseteq S} sum_A(r^*(Q)) \times P(Q), \tag{6}$$

where S is the set of information sources, r^* is the function represented by r (hence $r^*(Q)$ is the regular relation in the alternate world of r corresponding to $Q \subseteq S$,) and $P(Q)$ is the probability associated with Q (and with $r^*(Q)$). Note that

$$P(Q) = \prod_{s_i \in Q} p_i \prod_{s_i \notin Q} (1 - p_i).$$

Equation 6 can be written as

$$\sum_{Q \subseteq S} \sum_{t \in r^*(Q)} t(A) \times P(Q), \tag{7}$$

or

$$\sum_{t @ x \in r} t(A) \times \sum_{\mathcal{Q}} P(Q), \tag{8}$$

where x is the set of source vectors corresponding to the pure tuple in r, and $\mathcal{Q} = \{\mathcal{Q} | t \in r^*(Q)\}$.

We have shown in [12] that

$$re(t) = \sum_{\mathcal{Q}} P(Q).$$

In fact, this is the justification that reliability calculation algorithms are correct with respect to the alternate worlds semantics, i.e., the reliability calculated for a pure tuple t is exactly the sum of the probabilities of the alternate world relations where t appears. Hence Eq. 8 simplifies to Eq. 5. □

The P4-COUNT problem, i.e., finding the expected value of $count(r)$ of an extended relation r, is similar to P4-SUM. In fact, *count* can be implemented as the *sum* by adding an attribute A with a value of 1 to all the tuples, and then obtaining $sum_A(r)$. Below we will only give the algorithm. The proof of correctness follows from that of P4-SUM.

5.1.2. Algorithm P4-COUNT

Given an extended relation $r = \{t_1 @ x_1, t_2 @ x_2, , \ldots, t_n @ x_n\}$, where t_i is a pure tuple, and x_i is the set of information source vectors corresponding to t_i, the expected value of $count_A(r)$ can be calculated as

$$\sum_{i=1}^{n} re(t_i). \tag{9}$$

5.1.3. Normalized expected values

As discussed in the previous section, we might be interested in a *normalized* expected value of an aggregate operation, where the empty relation (if existent) in the alternate world on the extended relation of interest is regarded as irrelevant. To obtain a normalized expected value, we need the probability associated with the empty relation. The following algorithm can be used to obtain this probability.

5.1.4. Algorithm to compute the probability of the empty relation

The input to the algorithm is an extended relation $r=\{t_1@x_1,\ldots,t_n@x_n\}$. Let $\mathcal{Q}_\phi$ be the set of subsets of information sources giving rise to the empty relation, that is

$$\mathcal{Q}_\phi=\{Q|r^*(Q)=\Phi\}. \tag{10}$$

We can compute a source vector set that characterizes $\mathcal{Q}_\phi$ as follows:

$$x_\phi=(\overset{s}{\neg}x_1)\overset{s}{\wedge}(\overset{s}{\neg}x_2)\overset{s}{\wedge}\cdots\overset{s}{\wedge}(\overset{s}{\neg}x_n), \tag{11}$$

where $\overset{s}{\neg}$ and $\overset{s}{\wedge}$ are the source vector operations s-negation and s-conjunction.

The probability of the empty relation can be calculated from x_ϕ in the same manner that the probability of a tuple associated with x_ϕ is calculated (see Section 2.5). Let us denote this probability by p_ϕ. The normalized expected value is obtained by dividing the (nonnormalized) expected value by $1-p_\phi$.

It is easy to see why Eq. 11 correctly characterizes the set $\mathcal{Q}$, and we will only sketch the proof here. The expression corresponding to x_ϕ, $e(x_\phi)$, is true iff all the expressions corresponding to $x_1,\ldots,x_n$ are false, in which case all tuples $t_1,\ldots,t_n$ are invalid. Hence if $e(x_\phi)=true$ under $truth(Q)$, for a set $Q\subseteq S$ of information sources, then $r^*(Q)=\Phi$ is the empty relation.

5.1.5. Expected value of the average operation

A simple equation similar to Eqs. 5 and 9 cannot be obtained for the *average* operation. We might use the following definition for an approximate expected average, bearing in mind that this value is generally not equal to the actual expected average.

$$\text{approx-exp-}avg_A(r)=\frac{\text{exp-}sum_A(r)}{\text{exp-}count(r)}. \tag{12}$$

An efficient algorithm to calculate expected average (or proving the problem to be NP-complete) requires further investigation.

5.2. max and min operations

In this section we present an algorithm to determine, for a constant c, the probability of $max_A(r)=c$ (or $min_A(r)=c$). As we proved in Section 4, this problem, which we

called the P1 problem, is NP-complete for *sum*, *count*, and *average* operations. We will specify the algorithm for the *max* operation. The corresponding algorithm for the *min* operation can be obtained with slight modifications.

5.2.1. Algorithm P1-max: The probability of $max_A(r)=c$

Obviously, if c does not appear in r as an A-value, then the probability of $max_A(r)=c$ is zero. In the following, we assume that $c=t(A)$ for at least one pure tuple $t \in r$. As usual, assume $r=\{t_1@x_1, \ldots, t_n@x_n\}$. We first project r over A, and sort the result according to the A-values. (Note that in IST the information source attribute I is invisible to the users, hence the scheme of the result will be $\{A, I\}$.) Let $r'=\langle a_1@y_1, \ldots, a_m@y_m\rangle$ be the resulting sequence, where $a_1 > a_2 > \cdots > a_m$. Assume $c=a_j$ for some $1 \leqslant j \leqslant m$. For each A-value a_i we can calculate a set of information source vectors characterizing the relations in the alternate world of r in which a_i is the maximum. Let z_i denote this set of vectors for a_i. Then the probability of $max_A(r)=c$ can be obtained from z_i in the same manner that the probability of a tuple associated with z_i is calculated (see Section 2.5). The set of source vectors z_i is obtained as follows:

$$z_i = y_i \overset{s}{\wedge} \left(\overset{s}{\neg} \left(\overset{i-1}{\underset{k=1}{\overset{s}{\vee}}} y_k \right) \right), \tag{13}$$

where $\overset{s}{\neg}$, $\overset{s}{\wedge}$, and $\overset{s}{\vee}$ are the IST source vector s-negation, s-conjunction, and s-disjunction operations.

Theorem 10. *Given a constant c, Algorithm* P1-max *correctly determines the probability of* $max_A(r)=c$.

Proof. We need to show that z_i correctly characterizes the relations in the alternate world of r in which a_i is the maximum A-value. Consider the expression corresponding to z_i, $e(z_i)$. Let $Q \subseteq S$ be a subset of information sources such that $e(z_i)$ is *true* under *truth*(Q). Let $r^*(Q)$ be the regular relation in the alternate world of r corresponding to Q. We claim that (i) a_i appears in $r^*(Q)$, and (ii) No $a_k > a_i$ appears in $r^*(Q)$. These follow from the construction of z_i, Eq. 13. We have shown that the source vector s-conjunction and s-disjunction operations implement the logical conjunction and disjunction of the expressions corresponding to source vectors, respectively, and the source vector s-negation operation implements the logical negation. Since $e(z_i)$ is *true* under *truth*(Q), then so is $e(y_i)$. Hence a_i appears in $r^*(Q)$. Similarly, $e(y_k)$ is *false* under *truth*(Q) for $1 \leqslant k < i$, and hence no $a_k > a_i$ appears in $r^*(Q)$. □

Example 4. Consider the following extended relation *sample*, which is already ordered according to the A-values as seen in Fig. 5.

A	I		
80	1	−1	0
60	1	0	1
60	1	1	−1
50	0	1	−1
30	−1	1	1
30	1	0	−1

Fig. 5. The sample relation.

To find the probability of $max_A(sample) = 50$ we need to calculate

$$z = \{(0\ 1\ -1)\} \overset{s}{\wedge} (\overset{s}{\neg}\{(1\ -1\ 0) \overset{s}{\vee} (1\ 0\ 1) \overset{s}{\vee} (1\ 1\ -1)\})$$

obtaining $z = (-1\ 1\ -1)$. If, for example, the reliabilities of the information sources s_1, s_2, and s_3 were 90%, 80%, and 70%, respectively, we would obtain a probability of 2.4% for $max_A(sample) = 50$. □

Algorithm P1-max can also be used to enumerate the possible values of $max_A(r)$ and their probabilities for an extended relation r in the decreasing order. It is easy to show that

$$\overset{s}{\neg}\left(\overset{i}{\underset{k=1}{\overset{s}{\vee}}} (y_k)\right) = \overset{s}{\neg}\left(\overset{i-1}{\underset{k=1}{\overset{s}{\vee}}} (y_k)\right) \overset{s}{\wedge} (\overset{s}{\neg} y_i),$$

which provides an iterative algorithm to obtain the sequence of $z_1, z_2, \ldots$ using Equation 13. This approach can be efficient in applications where a few of the possible values is needed (as opposed to listing all possible values, which is exponential in the number of information sources in the worst case.)

5.2.2. Problems P2-max and P3-max

Algorithm P1-max can be used, in a straightforward way, to answer problem P'2-max (finding the *largest* maximum of an attribute value), and problem P'3-max (whether it is possible for the maximum to be greater than or equal to a given value).

We can also design algorithms for problems P''2-max and P''3-max based on P1-max. The algorithms are more involved than the previous cases, and are described below.

5.2.3. Problem P''2-max

We want to find the smallest possible value of $max_A(r)$ for an extended relation r. Let r' be the sorted relation as in Algorithm P1-max, that is, $r' = \langle a_1 @ y_1, \ldots, a_m @ y_m \rangle$, where $a_1 > a_2 > \cdots > a_m$. We could apply Algorithm P1-max to find the smallest a_i having a nonzero probability, by starting at a_m and

working in the increasing order of a_is. However, in the worst case, we may be forced to examine all $a_m, \ldots, a_1$. To improve the efficiency, we can use a technique similar to binary search: at each iteration, we have a search space of $\langle a_i, \ldots, a_{i+j} \rangle$, with a median element a_{i+k}. The question to answer is "It is possible for $max_A(r)$ to have a value less than or equal to a_{i+k}?" This is problem P"3-max, and its algorithm is presented below. If there are m A-values, $a_1, \ldots, a_m$, the worst case complexity of this algorithm is $\log m$ times the complexity of P"3-max algorithm. A "linear search" algorithm would have a worst case complexity of m times the complexity of P1-max algorithm.

5.2.4. Problem P"3-max

Given a value c, is it possible for $max_A(r)$ to be less than or equal to c? Let r' be the sorted relation as in Algorithm P1-max, that is, $r' = \langle a_1 @ y_1, \ldots, a_m @ y_m \rangle$, where $a_1 > a_2 > \cdots > a_m$. Special cases, where $c > a_1$, $c = a_1$, or $c < a_m$ are simple to handle. So, assume without loss of generality, that $a_i > c \geqslant a_{i+1}$, for some $i = 1, \ldots, m-1$. We can form a set of source vectors z that characterize the probability that $max_A(r) \leqslant c$ as follows:

$$z = \overset{s}{\neg}\left(\overset{i}{\underset{k=1}{\overset{s}{\vee}}} y_k \right) \overset{s}{\wedge} \left(\overset{m}{\underset{k=i+1}{\overset{s}{\vee}}} y_k \right), \tag{14}$$

where $\overset{s}{\neg}$, $\overset{s}{\wedge}$, and $\overset{s}{\vee}$ are the IST source vector s-negation, s-conjunction, and s-disjunction operations.

In fact, if all source vectors of z contain the $\top$ element, then the answer to the problem is negative. Otherwise, if z contains at least one source vector with no $\top$ element, then it is possible that $max_A(r)$ has a value less than or equal to c. The corresponding probability can be obtained from z using a reliability calculation algorithm of [11] (See Section 2.5).

5.2.5 Problems P1-min, P2-min, and P3-min

Algorithm P1-min, finding the probability of $min_A(r) = c$. is very similar to P1-max. The only difference is that the projection of r onto attribute A is sorted to obtain r' in the nondecreasing order of A values. That is, $r' = \langle a_1 @ y_1, \ldots, a_m @ a_m \rangle$, where $a_1 < a_2 < \cdots < a_m$, and each y_i, $i = 1, \ldots, m$, is a set of source vectors. Then, formula 13 characterizes the probability that $min_A(r) = a_i$.

As for the maximum operation, variants of Algorithm P1-min can be used to answer problems P2-min and P3-min. The algorithms are similar to those of the maximum operation, and are omitted.

6. Summary and conclusions

We discussed the processing of aggregate operations *min*, *max*, *sum*, *count*, and *average* in database systems with uncertain and inaccurate information. The

Information Source Tracking method, IST, was used for the modeling and manipulation of uncertain information. When uncertainty is present, the answer to an aggregate query is not unique.

We presented an algorithm to find all the answers to an aggregate query. This algorithm has a $O(2^k \times n)$ complexity, where k is the number of information sources, and n is the size of the relation.

Then we studied a number of other formulations, namely, for an extended relation r, an aggregate operator *agg* and an attribute A,

(P1) Given a constant c, is the probability of $agg_A(r) = c$ nonzero?

(P2) Find the largest (or smallest) possible value of $agg_A(r)$.

(P3) Given a constant c, is it possible that $agg_A(r) \geqslant c$ (or $agg_A(r) \leqslant c$)?

(P4) Find the expected value of $agg_A(r)$.

We proved that problems P1, P2, and P3 are NP-complete for the aggregate operators *sum*, *count*, and *average*. Algorithms were presented for expected values (problem P4) of *sum*, and *count* queries, as well as for the *min* and *max* versions of problems P1, P2, and P3. The complexities of problem P4 are still unknown for the *min*, *max*, and *average* operations. An approximate value for expected average can be obtained as the fraction of expected sum divided by expected count.

Some other questions are possible with respect to aggregate queries that merit further investigations. One such problem is to find the most likely answer to an aggregate query, i.e., the answer with the highest probability.

Acknowledgements

The author wishes to thank anonymous referees for helpful comments. This work was supported by grants from the Natural Sciences and Engineering Research Council of Canada (NSERC), and Fonds pour la Formation de Chercheurs et l'Aide a la Recherche (FCAR) of Quebec.

References

[1] J. Biskup, A foundation of Codd's relational maybe operations, *ACM Trans. Database Systems* **8** (4) (1983) 608–636.

[2] S.A. Cook, The complexity of theorem proving procedures, in: *Proc. 3rd ACM Symp. Theory of Computing* (1971) 151–158.

[3] L. DeMichiel, Resolving database incompatibility: An approach to performing operations over mismatched domains, *IEEE Trans. Knowledge Data Engrg.* **1** (4) (1989) 485–493.

[4] B. Doyon, Reliability of answers to an SQL query, Project Report, Department of Computer Science, Concordia University, May 1990.

[5] M.R. Garey and D.S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-Completeness* (Freeman, SanFransisco, CA, 1979).

[6] A.M. Keller and M. Winslett-Wilkins, On the use of an extended relational model to handle changing incomplete information, *IEEE Trans. Software Engrg.* **sE-11**(7) (1985) 620–633.

[7] K.-C. Liu and R.Sunderraman, Indefinite and maybe information in relational databases, *ACM Trans. Database Systems* **15** (1) (1990) 1–39.
[8] K.-C. Liu and R. Sunderraman, A generalized relational model for indefinite and may be information, *IEEE Trans. Knowledge Data Engrg.* **3** (1) (1991) 65–77.
[9] D. Maier, *The Theory of Relational Databases* (Computer Science Press, Rockville, MD, 1983).
[10] Nils J. Nilsson, Probabilistic logic, *Artificial Intelligence* **28** (1986) 71–87.
[11] F. Sadri, Reliability of answers to queries in relational databases. *IEEE Trans. Knowledge Data Engrg.* **3** (2) (1991) 245–251.
[12] F. Sadri, Modeling uncertainty in databases, *IEEE Internat. Conf. Data Engrg.* (1991) 122–131.
[13] F. Sadri, Integrity constraints in the information source tracking method, IEEE Trans Knowledge Data Engrg., to appear.
[14] F. Sadri, Information source tracking method: Efficiency issues, manuscript, submitted.